**Proyecto #1**

**Hospital Rush**

**Introducción a Programación**

**Profesor**

**Alejandro Alfaro**

**Alumno**

**Roney Ruiz Rojas**

**Agosto 06, 2025**

**Arquitectura General**

El proyecto está organizado en módulos siguiendo una estructura MVC simplificada y orientada a componentes.

* **src/Components/**: Contiene los componentes visuales y lógicos del juego (escenarios, personajes, botones, drag & drop, equipamiento).
* **src/Models/**: Define modelos de datos como coordenadas, tamaño y partidas históricas.
* **src/Helpers/**: Funciones auxiliares para manejo de assets y partidas.
* **src/Skills/**: Habilidades de personajes y su gestión.
* **src/Constants/**: Constantes globales (assets, detalles de pantalla).

La jerarquía de componentes se basa en herencia y composición. Por ejemplo, todos los personajes heredan de BaseCharacter.py, y los botones de BaseButton.py.

**Problema**

El juego simula la gestión de un hospital en situaciones de emergencia, donde el jugador debe tratar pacientes, gestionar recursos (camillas, doctores) y responder a eventos dinámicos bajo presión de tiempo.

El ciclo principal está en el main.py, donde se procesan eventos y se actualizan/dibujan los escenarios.

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

**Meta**

El objetivo es tratar la mayor cantidad de pacientes posible antes de que se acabe el tiempo, utilizando las habilidades de los personajes y gestionando la fatiga y reparación de equipamiento.

La meta se verifica en el método end\_game del RoomScenario.py, donde se calcula la cantidad de pacientes tratados y se registra la partida.

A screen shot of a computer code

AI-generated content may be incorrect.

**Usuarios y Roles**

El usuario controla doctores y enfermeras, cada uno con habilidades específicas. Los roles están definidos por clases como DoctorCharcter, TriageCharacter y PatientCharacter.

Cada personaje tiene habilidades que se asignan al instanciarse y se usan según el contexto (por ejemplo, reparar camillas o tratar pacientes).

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.A screen shot of a computer code

AI-generated content may be incorrect.

**Reglas de Juego**

* Los doctores pueden tratar pacientes si tienen las habilidades requeridas.
* Las camillas se desgastan y requieren reparación.
* Los doctores se fatigan y deben recuperarse.
* El tiempo es limitado

**Eventos Dinámicos**

Los eventos incluyen el desgaste de camillas, la fatiga de doctores y la aparición de pacientes con diferentes niveles de gravedad.

Todo esto se desarrolla mediante propiedades de clases como el nivel de fatiga de los doctores y los usos de las camillas. Se evalúan comportamientos mediante condiciones según esas mismas propiedades y condiciones están pendientes escuchando el bucle de eventos.

**Condiciones de Victoria y Puntaje**

Se gana si todos los pacientes son tratados antes de que termine el tiempo. La puntuación es la cantidad de pacientes tratados. Después de cada oleada se van guardando los datos en el historial y se muestra un mensaje según el resultado.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a video game

AI-generated content may be incorrect.

**Indicaciones Generales**

* El usuario interactúa arrastrando personajes y haciendo clic en botones.
* El estado del juego se reinicia al cambiar de escenario.
* El historial de partidas se muestra en el escenario del historial.

**Arquitectura de Comunicación**

El proyecto utiliza una arquitectura basada en componentes y escenarios, donde cada elemento del juego (botones, personajes, equipamiento, etc.) es un componente que puede escuchar y responder a eventos. Los escenarios agrupan y coordinan estos componentes, gestionando el flujo del juego.

* **Escenarios**: Cada escenario (por ejemplo, menú, sala de juego, historial) extiende de BaseScenario.py y tiene métodos para escuchar eventos y dibujar en pantalla.
* **Componentes**: Los componentes como botones, personajes y equipamiento heredan de Component.py y pueden implementar los métodos listen y draw para interactuar con el usuario y el entorno.

A screen shot of a computer program

AI-generated content may be incorrect.

**Ejemplo de Interacción entre Componentes**

* **Drag & Drop**: Los personajes (DoctorCharacter.py, PatientCharacter.py) implementan la interfaz Draggable.py, permitiendo que el usuario los arrastre y los coloque en zonas específicas (DropZone.py).
* **Interacción en camillas**: Cuando un doctor y un paciente están en la misma camilla, el componente Interaction.py valida si el doctor puede tratar al paciente y gestiona el proceso de tratamiento.

**Nota:** Todo el tema del Drag & Drop es algo complejo y si desea ver más detalles le sugiero revisar los archivos para mayor comodidad.

**Repositorio del Proyecto:** <https://github.com/ronz204/isw221-hospital-rush>

El proyecto es publico y yo ya le había mandado una invitación como colaborar así que con aceptarla ya tiene toda la disponibilidad de clonarlo e indagar sobre este proyecto.